Attack-tree based risk analysis of Estonian i-voting

Summary for Research Seminar in Cryptography

Riivo Talviste

December 15, 2014

1 Introduction

This report analyzes two independent works published in 2014 that model security threats of Estonian i-voting scheme using attack trees. The first one, the master's thesis of Tanel Torn [11] constructs several realistic attack trees for various types of attacks on Estonian i-voting system and evaluates them using three different state-of-the-art methodologies proposed in attack-tree literature. The second work, the master's thesis of Ruud Verbij [13], proposes a general framework to allow comparison of different internet voting schemes. Verbij evaluates the proposed framework by applying it on Estonian i-voting protocol.

2 Attack trees

When analyzing security of a system it may be difficult to estimate different properties of an attack. For example, how much does "malware attack to modify electronic votes" cost or how probable it is? One way to answer these questions is to model the given attack as an attack tree. Attack tree takes the ultimate attack goal and divides it recursively into subattacks in a tree like fashion until the leaf nodes of the tree are elementary attacks that are easier to assign properties to. Figure 1 presents a simple example with the main goal of getting a free lunch.

Conceptually, there are two kinks of nodes in the attack tree: OR-nodes, that are satisfied if at least one of their children is executed; and AND-nodes (shown with arcs in Figure 1), that are reached if all their child nodes are executed.

Although the concept of attack trees had been used beforehand, the term was coined by Schneier in 1999 [9].

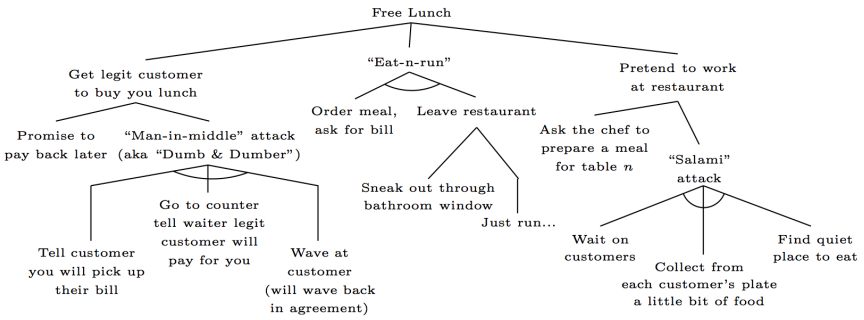

Figure 1: A simple attack tree. Originally published in [7].

2.1 Evaluating attack trees

Remember, that the purpose of using attack trees is to evaluate some properties (e.g. monetary cost) for the root node of the tree by splitting the attack into smaller parts that are easier to evaluate. The leaf nodes (elementary attacks) can be evaluated using information from published papers, statistics or even common sense.

In the simplest case the cost of parent node is calculated by linearly adding up the costs of its child nodes. However, exist are more involved ways of evaluating attack trees. In his work, Torn uses multi-parameter attack trees that use more than one attribute to describe each node, for example, cost, gains, success and failure probabilities, expected penalties, etc. He uses three different methodologies to evaluate such attack trees. Two of them ([2] and [4]) can be considered parallel models, where possible subattacks are evaluated in parallel; and one is a serial model [5], where the outcome of each subattack are taken into account to make the next decision.

3 Internet voting in Estonia

In this work we consider Estonian electronic voting system that allows voters to vote over the internet. Hence, we call it i-voting rather than e-voting, although in Estonian context they are the same.

Estonian i-voting scheme uses a so-called "double envelope" protocol that resembles voting by postal service. Overview of the process is shown in Figure 2.

First, the voter downloads the voting application and using it, authenticates herself to the Vote Forwarding Server (VFS) using either ID-card (compulsory for Estonian citizens) or (supplemental) mobile-ID. The VFS verifies that the voter is eligible to vote and checks from the Vote Storage Server (VSS) if the voter has already voted. Voters are allowed to cast their i-vote any number of times, overriding previous one. The latter check is just for informing the voter about the existence of previous vote. If the voter is eligible to vote, VFS replies

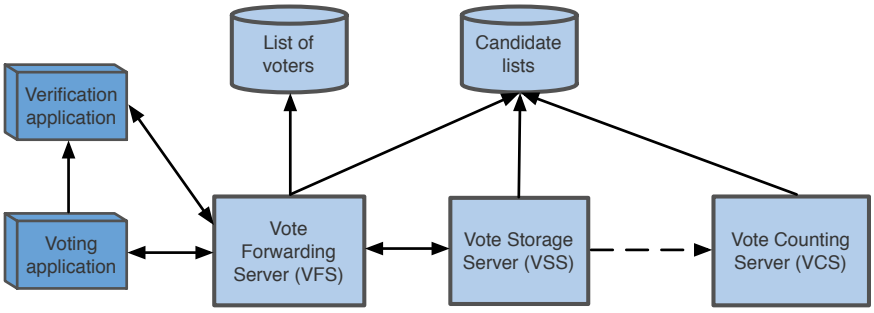

Figure 2: Overview of Estonian i-voting system. Figure adapted from [12].

with the candidate list corresponding to the voter's constituency.

For each elections, a new key pair is generated by the National Electoral Committee (NEC) that oversees the elections. The key pair is generated by an hardware security module connected the the offline Vote Counting Server and the corresponding secret key (sk_{NEC}) never leaves the device. A successful authentication (using key cards and PIN codes) of the majority of the 7-member key managers group is required to perform any operations with the secret key. The elections' public key (pk_{NEC}) is embedded in the voting application.

During the voting process, the voter's choice (candidate number), together with generated randomness, is encrypted using the public key of the elections. The randomness is required so that different encryptions of the same candidate are indifferentiable. This encryption forms the first envelope. This encryption is then signed by the voter using either ID-card or mobile-ID. The signature forms the second, outer envelope. Hence, the vote is structured as this:

$$\mathsf{Sig}_{sk_{voter}}\mathsf{Enc}_{pk_{NEC}}(Vote, Rnd),$$

where sk_{voter} is the private key of the voter, $Vote$ is the chosen candidate number and Rnd is the randomness. The encrypted and signed vote is sent to the Vote Forwarding Server which in turn sends it to the Vote Storage Server. The latter checks the validity of the voter's signature and a confirmation is sent back to the voter. VSS also invalidated the voter's previous votes is they exist.

After the i-voting period ends, a list of i-voters is compiled by VSS and sent to the polling stations. If a voter has voted with both i-voting and paper ballot during the advance voting period, the paper ballot takes priority and the i-vote is invalidated. Remaining valid i-votes are stripped of the signatures and anonymous encryptions are transferred to Vote Counting Server (VCS) using offline data carrier (optical disk), where the elections' private key is used to count the votes and publish the tally.

3.1 Verification process

As of 2013, voters are able to verify is their i-vote has reached the Vote Storage Server as intended. This verification method is implemented mainly to discover

any malware present in the voter's computer that modifies the vote. Hence, an independent device must be used for verification. In the current scheme this device it chosen to be a smart device (smartphone or tablet computer) with a camera and internet connection. Overview of the process is given in Figure 3.

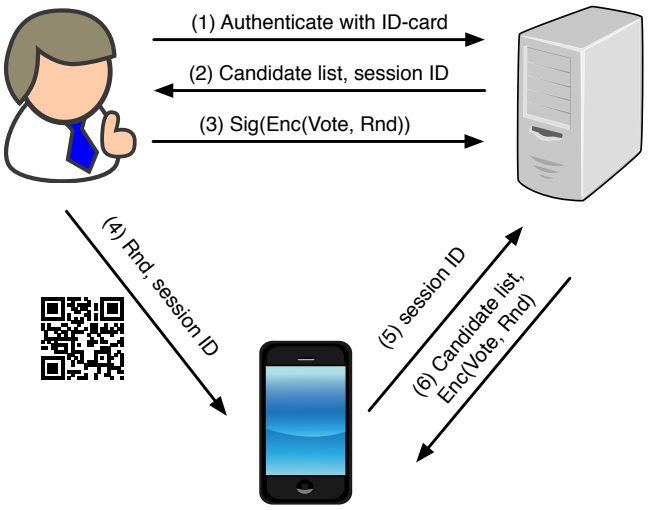

Figure 3: Overview of vote verification. The first three steps are the vote casting process and remaining three the verification process. Figure adapted from [12].

After a successful voting process, the voting application displays a QR code embedded with the randomness used to encrypt the vote and a session ID. The latter is generated by the VFS and used as a unique identifier for a particular voting process. To verify her vote, the voter can download the verification application to her smart device and use it to scan the QR code on the computer screen. The verification application decodes the session ID and randomness and uses the session ID to ask VFS for the corresponding vote. VFS replies with an anonymized vote (only the encryption without signature) and the corresponding candidate list. The verification application encrypts all the candidates in that list with the randomness used in the actual voting process and compares the result with the encryption sent by VFS. Upon match, the corresponding candidate name is shown to the voter. If no match is found than it means that the initial vote has been corrupted and the voter is directed to contact the National Electoral Committee.

4 Attacks on Estonian i-voting

As described, Estonian i-voting process can be divided into two parts — the actual casting of the vote in voter's computer and then everything else that happens before or after that: setting up voting servers, building and deploying the voting application and other software components, managing candidate lists,

receiving and saving votes and finally counting the votes. The latter part can be referred to as the Central System.

Attacks on Estonian i-voting system can be categorized similarly. Attacks on the actual voting process mean attacking the voter's computer and this is done by either a fake voting application or other malware that intercepts the voting process or casts votes on behalf of the legitimate voter. Attacking the Central System means either infecting the servers with a specially crafted malware or bribing electoral officials, auditors, IT staff or other people that have access to the critical infrastructure. In the following, attacks proposed by Torn [11] and Verbij [13] are divided into these two categories.

4.1 Attacks on Central System

Torn considers four ways to attack the Central System:

- Compromise Vote Forwarding Server (VFS) — [11] gives no details here, but it can be assumed that the process would be similar to attacking the VSS.
- Compromise Vote Storage Server (VSS) — change the list of votes on VSS before they are carried to VCS. An attacker would need to place malware in VSS by either bribing an employee or hacking into the server.
- Compromise Vote Counting Server (VCS) — change the vote counting algorithm or other software in VCS. An attacker would need to bribe an employee. Hacking into VCS is not feasible as this server is offline.
- Compromise data carrier — modify votes while they are transferred from VSS to VCS. The fastest and and less detectable way would be to replace the original data carrier with a fake one that has an alternative list of anonymous votes on it. Doing so would require bribing or infiltrating as an employee or an observer.

Torn also describes a vote publishing attack, where an attacker first gets the encryptions of votes by breaking into VSS and then obtains the private key of the elections. The author of this report deems the latter practically impossible as the private key resides in a hardware security module.

In his work, Verbij considers bribing or coercing to be exponentially harder for each subsequent individual. Thus, in his framework, it is considered impossible to bribe or coerce more than one member of either the electoral authorities, IT staff or auditors without getting caught. Since all these positions are held by two or more people, successfully bribing someone is considered difficult.

In his framework in general, Verbij considers hacking into voting infrastructure (servers) or manipulating communication network. It can be assumed that voting servers are monitored with care to spot any suspicious activity. Hence, the author concludes that voting infrastructure can only be attacked using zero-

day vulnerabilities[1]. As for network, most of the man-in-the-middle attacks are mitigated by the use of authenticated TLS connections. However, the author notes that SMS messages (that are for example used in Norwegian e-voting) can be faked to seem to be sent by election authorities.

Both Torn and Verbij also describe an attack on availability of the voting system where an attacker uses a botnet to launch a distributed denial of service (DDoS) attack on the voting servers during the election period. Torn proposes that to use a DDoS attack for revoking the election results, the voting servers would have to be inaccessible for at least 80% (5 days) of the online voting time. He estimates that this would cost 6300 €[2]. Verbij estimates that taking down the voting servers with DDoS for one day costs $7980, however, this includes botnet infrastructure cost subsequent days will cost less. Verbij relies on underground economy research in [3, 10, 6].

4.2 Attacking voters' computers

As attacking the Central System proves to be costly and involves high risk, both Torn [11] and Verbij [13] concentrate on attacks against voters and their computers. They consider two kinds of vote modification attacks. Firstly, both authors analyze attacks where the attacker modifies legitimate votes to be cast for another preferred candidate. The goal here is to get more legitimate-looking votes for the preferred candidate and so the attack itself has to go undetected. Secondly, both authors consider a revocation and/or reputation attack against the concept of i-voting with the goal to invalidate the election results and ultimately ban the i-voting altogether. The attack vector here is similar to the previous one, however, the attack could and even should be detected so it could be used as a proof that i-voting is vulnerable to attacks. Not having to keep the attack secret, makes it easier and also cheaper for the attacker.

Torn considers vote manipulation attack through malware and details three types of malware:

- Vote Modifying Malware — a malware that either changes (Vote Changing Malware) or blocks (Vote Blocking Malware) the legitimate vote during the i-voting process. Also needs a corresponding malware present in user's smart device to fake verifying process.
- Re-voting malware — a malware that changes the original vote by re-voting on behalf of the voter after some amount of time. The malware has to save user's ID-card PIN during the initial voting process and ID-card has to be left in the card reader for some time.
- Self-voting Malware — a malware that just votes on behalf of a user any time during the online voting period. User's ID-card has to be in the card reader and malware has to have (previously) learned user's PIN codes.

[1] Verbij relies on a study published in 2014 [1], where a zero-day vulnerability is estimated to cost $20 000 – $50 000 for Mac OS X and $60 000 – $100 000 for Windows.

[2] This includes 3000 € for convincing the NEC. Unfortunately, Torn does not give any references here.

In all cases, Torn takes as a goal to successfully change 5000 votes as this would guarantee one extra seat in Estonian parliament. As Torn uses multi-parameter attack trees in his work, he tries to attach realistic numeric values to the success probabilities, penalties, costs and gains for leaves of the attack tree. An overview of chosen values is given in Tables 1 and 2.

Table 1: Estimated probabilities needed for constructing the attack tree.

Action or event	Probability
Overall success rate of the attack:	0.8
Successful development of malware:	0.95
Probability for getting caught for developing malware:	0.05
Detection of large-scale network infection:[a]	0.95

[a] Torn estimates that NEC has the has high cooperation with Estonian CERT and so large-scale network attacks are easily detected in Estonia [11].

Table 2: Property values used for assigning values to the attack tree.

Property	Value
Votes needed to be changed:[a]	5000
Spending one year in prison:[b]	131 400 €
Imprisonment for developing malware:[c]	3 years
Imprisonment for distributing malware:[d]	9 years
Hourly rate for attacker:	50 €
Cost of botnet of size N:[e]	$N/100000 \cdot 26000 + 20 \cdot 8 \cdot 50$
Creating the malware:[f]	4 weeks + 2 days days

[a] Equals one seat on Estonian parliament [11].

[b] Given that on a normal legal job working 8 hours a day, an attacker earns 15 €/hour [11].

[c] Torn refers here to the Estonian Penal Code §216.

[d] Torn refers here to the Estonian Penal Code §208 and §163.

[e] Torn refers to [8] that states that a botnet with 100 000 nodes was put on sale for 36 000 USD (˜26 000 €). Torn adds a month worth of attacker work to find the seller and set botnet up.

[f] This time consists of preparations before the i-voting period (4 weeks) and actual malware creation once the official application is published (2 days). Torn estimates, that the attacker works with double load on these two days to save important online time [11].

One may assume that to change 5000 votes, it is necessary to infect 5000 computers. However, there are many modifiers to take into account and as seen from Table 3, the number of devices to be infected for the Vote Changing Malware can increase drastically. For Vote Blocking Malware, no exact number is given in [11], but it is predicted to be 7 times the size of VCM. This

makes this kind of attack unrealistic in Estonian context. Similarly, for Revoting Malware the number of infected computers is calculated to be 218 000, which is again unrealistic. For Self-voting Malware this number is only 27 000. However, the author points out that in the case of SVM, the attack is almost certainly detected. This is due to two reasons. First, if a voter would try to vote electronically after a SVM has voted on her behalf, the voter would be notified about the existence of a previous vote. Secondly, in Estonia, voters who vote during advance voting period (this includes i-voting period) are not allowed to vote on the Election Day. So if the SVM has voted for somebody who would like to vote on the Election Day, then the foul play would be discovered.

Table 3: Calculation of the needed botnet size and its cost.

Property	Ratio
Votes needed to be changed:	5000
Computers belonging to eligible voters:[a]	0.9
Voter turnout (2011):	0.635
i-voters among voters (2011):	0.243
Votes not already cast for the preferred party:[a]	0.9
Overlap between infected computers and mobile devices:[a]	0.7
Malware active:	5 out of 7 days
Computers per household:[a]	1.5
Computers needed to be infected:	53 000
Cost for such botnet:	22 000 €

[a] Estimated by Torn [11] without any references to source.

In his thesis, Torn also considers a revocation attack that is similar to the vote modification attack through malware. The difference here is that the attacker does not have to worry about the vote verification part and does not have to make the attack undetectable. Moreover, instead of 5000 votes, the author estimates that changing 1000 votes is enough to ask for revocation of the election results.

In his work, Verbij considers a similar attack where botnet is used to distribute malware that changes votes. However, he takes a different approach on verification process. Instead of creating a separate malware for mobile devices, Verbij proposes a Vote Changing Malware that at the end of the voting process, displays a QR code that reuses session ID and randomness from some other legitimate vote that was previously cast for the candidate. These session ID and randomness pairs are collected and distributed by the botnet Command & Control nodes (see Figure 4).

Interestingly, Verbij fails to take into account the fact that the vote verification process is limited[3]: a single vote can only be verified in the next 30 minutes after it is cast and a maximum of three times. This limitation does not make

[3]The author of this report assumes that this is due to the lack of up-to-date documentation about Estonian i-voting system in English.

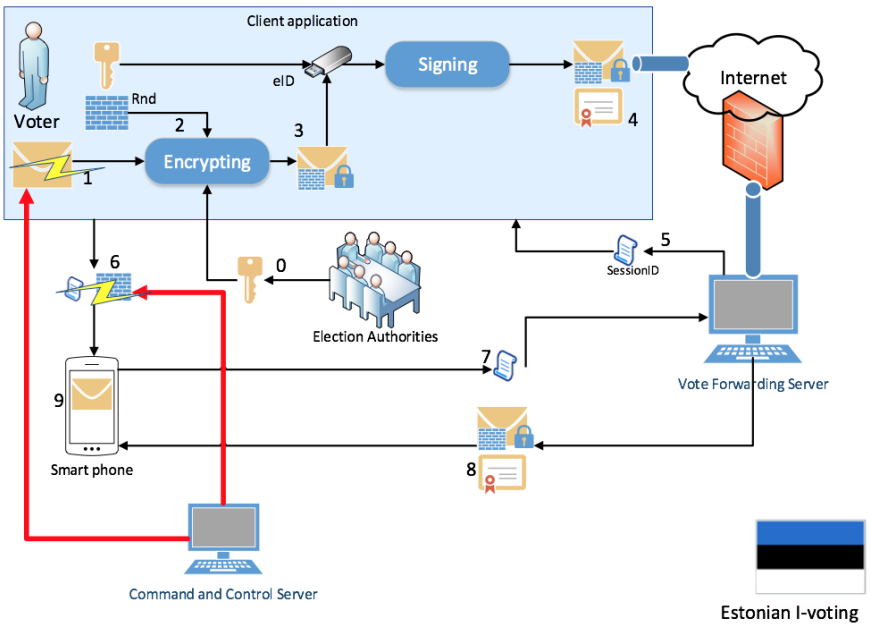

Figure 4: Malware attack that reuses valid session identificators [13].

the attack described in [13] impossible, but certainly makes the collection and distribution of session ID and randomness pairs more complicated.

Verbij does not assign success probabilities or expected outcomes to the elementary attacks and concentrates only on direct monetary costs and the time it takes to manifest the attack (see Table 4). Verbij considers the attacker to be working for free.

In his analysis, Verbij takes as a goal that in order to gain one seat in Estonian parliament, it is necessary to change 6000 votes[4]. Similarly to Torn, Verbij considers many factors contributing to the total number of computers needed for the attack. These factors are summarized in Table 3.

Figure 5 shows a decorated attack tree for this vote manipulation attack. The bottom row in each node represents the necessary time investment to fulfill that node and is split as follows: *active advanced* | *passive advanced* | *active actual* | *passive actual*. Interestingly, Verbij's estimate on the development of the malware and attack setup (5 days) is less favourable for the attacker than similar estimate (2 days) in Torn's work.

Like Torn, Verbij also considers another, simpler attack with the goal to diminish the reputation of i-voting and/or revoke the election results. The proposed attack is mostly like the previous attack for changing votes, but instead of changing the votes the malware sends the (yet unencrypted) vote together

[4]Based on the statistics of elections of 2011: `http://en.wikipedia.org/wiki/Estonian_parliamentary_election,_2011`

Table 4: Time constraints for the attacker model [13, page 87]

Task	Advance or actual voting days	Active time in days	Passive time in days
Decompiling voting app and building fake version	Actual	4-5	0
Decompiling voting app and inserting vote listener	Actual	2	0
Buy 30 000 nodes for botnet in one country	Advance	0	1
Deploy the actual malware to placeholder malware	Actual	0	1
Botnet consultation	Advance	1	0
Refining standard botnet code	Advance	2	0

with the voter's ID to the Command & Control servers that will periodically publish those votes. Hence this is an attack against the confidentiality of the votes. Verbij calculates that in order to publish one vote every hour, a total of 1,961 computers would need to be infected, making this attack cost only $3,255. The decorated attack tree for this confidentiality attack is shown in Figure 6.

4.2.1 Evaluating attack trees

Torn and Verbij take a different approach on evaluating the attack trees. As seen from the decorated attack trees (Figures 5 and 6), Verbij only considers direct costs and time constraints and adds these up linearly to get an evaluation for the root node.

Torn evaluates all of the attack trees proposed in his thesis with three different computational models described in Section 2.1. For the BLPSW model [2], Torn used an application of CoCoViLa[5], which is a model-based software development platform. However, since the parallel [4] and serial [5] models are computationally extensive, the author decided to split the attack trees into subtrees and evaluate them separately.

Torn calculates the outcome of each possible attack with gains varying from one to 20 million euros with steps of one million euros. As a result, for the vote manipulation attack, the expected outcomes are non-positive for all of the attack suites. On the other hand, the simpler vote revocation attack is more profitable. For example, with gains set to 4 million euros, the expected outcome of the revocation attack with all three models is about two million euros for the Self-voting Malware and 1.5 - 2 million euros with Vote Modifying Malware. Torn notes that it is important to keep in mind that such revocation attack is only beneficial to the parties who receive proportionally less i-votes then regular

[5]CoCoViLa — Model-Based Software Development Platform: `http://www.cs.ioc.ee/cocovila/`

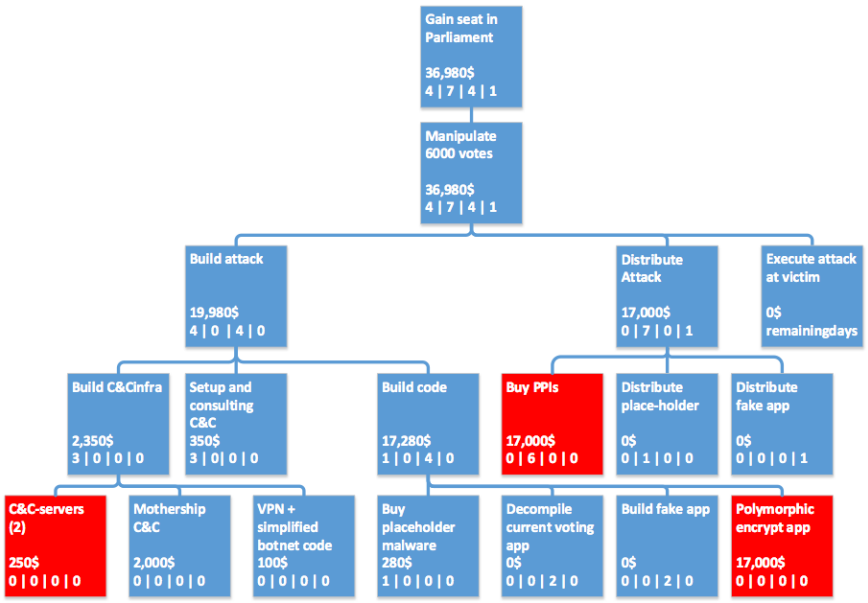

Figure 5: Decorated attack tree for the vote manipulation attack [13]

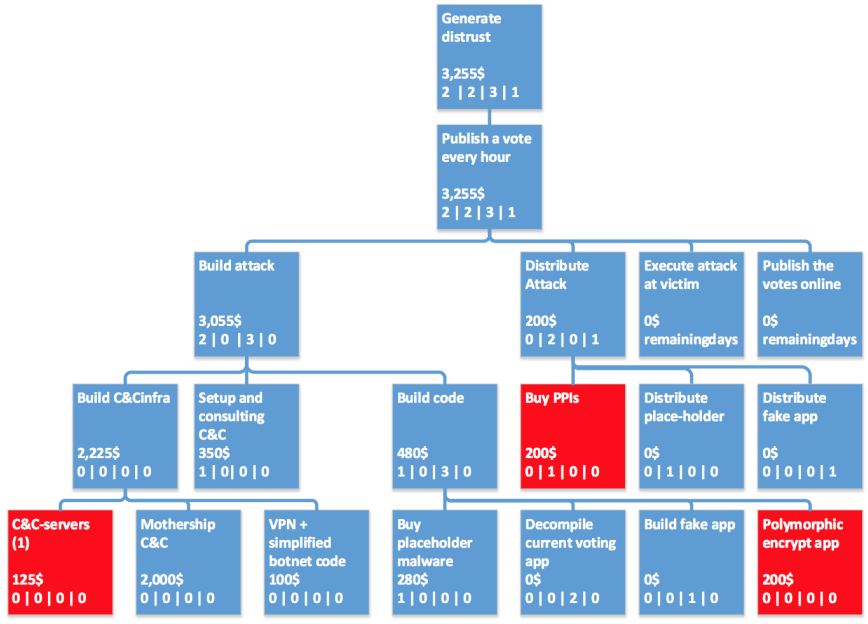

Figure 6: Decorated attack tree for the vote publishing attack [13]

Table 5: Calculation of the needed botnet size and its cost.

Reason	Modifier
Votes to modify:	6000
Computers used for i-voting:[a]	12.99%
Add a margin:	20%
Malware active:	2 days out of 7
Votes during last 2 days:[a]	36,43%
Add a margin:	10%
Computers to be infected:	170 000
Cost of botnet (USD):[b]	17 000

[a] Verbij [13] refers to official published statistics about i-voting in 2011: `http://www.vvk.ee/voting-methods-in-estonia/engindex/statistics`

[b] Verbij refers to a research paper by Trend Micro Inc. [3] and estimates that on average 1000 infected computers in a botnet cost $100. Note that this does not include the cost of Command & Control servers.

paper votes.

Torn also evaluates the attack trees for attack against the Central System described in Section 4.1. As expected, attacking the Central System is costly. Compromising data carrier yields negative outcome using all attack tree evaluation methodologies and only serial model [5] gives positive outcome for attacks compromising VFS, VSS or VCS. However, even then the outcomes are lower than those of malware attacks and so the author assumes the attacks on Central System to be less probable that those targeting users and their computers.

5 Conclusion

In their thesis, both Torn [11] and Verbij [13] analyze the security of Estonian i-voting scheme by constructing possible attacks against either the confidentiality, integrity of the votes or availability of the system. They use attack trees to model and evaluate the proposed attacks. However, the authors take different approach while doing so. Torn tries to estimate the possible expenses and penalties for the attacker to see if launching a given attack suite would be profitable for the attacker for a given gain. Verbij takes a simpler approach and takes into account only direct costs and time limitations. He considers all subattacks to succeed with probability 100%. Consequently, Verbij's proposed attacks yield frightening results and can be considered best-case scenarios for the attacker.

Despite using different approaches, both Torn and Verbij agree on some of the results. First, they both consider attacks on the Central System to be much more expensive, involving more risk and thus less probable. Second, results of both authors' analyzes show that revocation attacks are more profitable than vote modification attacks. This in mainly due to the fact that in the former case the attack does not have to go through undetected.

References

[1] Lillian Ablon, Martin C Libicki, and Andrea A Golay. *Markets for Cybercrime Tools and Stolen Data: Hackers' Bazaar*. Rand Corporation, 2014.

[2] Ahto Buldas, Peeter Laud, Jaan Priisalu, Märt Saarepera, and Jan Willemson. Rational Choice of Security Measures Via Multi-parameter Attack Trees. In *Critical Information Infrastructures Security*, volume 4347 of *LNCS*, pages 235–248. Springer, 2006.

[3] Max Goncharov. Russian underground 101. Trend Micro Incorporated Research Paper, 2012.

[4] Aivo Jürgenson and Jan Willemson. Computing Exact Outcomes of Multi-parameter Attack Trees. In *On the Move to Meaningful Internet Systems: OTM 2008*, volume 5332 of *LNCS*, pages 1036–1051. Springer, 2008.

[5] Aivo Jürgenson and Jan Willemson. Serial Model for Attack Tree Computations. In *Information, Security and Cryptology – ICISC 2009*, volume 5984 of *LNCS*, pages 118–128. Springer, 2010.

[6] Derek Manky. Cybercrime as a service: a very modern business. *Computer Fraud & Security*, 2013(6):9–13, 2013.

[7] Sjouke Mauw and Martijn Oostdijk. Foundations of Attack Trees. In *Information Security and Cryptology – ICISC 2005*, volume 3935 of *LNCS*, pages 186–198. Springer, 2006.

[8] Yury Namestnikov. The economics of botnets. Published online at `http://securelist.com/large-slider/36257/the-economics-of-botnets/`, 2009.

[9] Bruce Schneier. Attack trees. *Dr. Dobbs journal*, 24(12):21–29, 1999.

[10] Aditya K Sood and Richard J Enbody. Crimeware-as-a-service – a survey of commoditized crimeware in the underground market. *International Journal of Critical Infrastructure Protection*, 6(1):28–38, 2013.

[11] Tanel Torn. Security analysis of Estonian i-voting system using attack tree methodologies. Master's thesis, Tallinn University of Technology, 2014.

[12] Vabariigi Valimiskomisjon. Elektroonilise hääletamise süsteemi üldkirjeldus. Published online at `http://vvk.ee/public/dok/elektroonilise-haaletamise-systeemi-yldkirjeldus-EH-03-03-1_2013.pdf`, 2013. In Estonian.

[13] Ruud Verbij. Dutch e-voting opportunities. Master's thesis, University of Twente, 2014.